Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2021.012230

ARTICLE

A Trusted NUMFabric Algorithm for Congestion Price Calculation at the Internet-of-Things Datacenter

Shan Chun[1], **Xiaolong Chen**[2], **Guoqiang Deng**[3,*] **and Hao Liu**[4]

[1]School of Electronics and Information, Guangdong Polytechnic Normal University, Guangzhou, 510665, China

[2]Information Engineering College, Jinhua Polytechnic, Jinhua, 32100, China

[3]The Information Network Engineering and Research Center, South China University of Technology, Guangzhou, 510641, China

[4]Qianxin Technology Group Co., Ltd., Beijing, 100044, China

[*]Corresponding Author: Guoqiang Deng. Email: Denggq@yeah.net

Received: 21 June 2020 Accepted: 16 October 2020

ABSTRACT

The important issues of network TCP congestion control are how to compute the link price according to the link status and regulate the data sending rate based on link congestion pricing feedback information. However, it is difficult to predict the congestion state of the link-end accurately at the source. In this paper, we presented an improved NUMFabric algorithm for calculating the overall congestion price. In the proposed scheme, the whole network structure had been obtained by the central control server in the Software Defined Network, and a kind of dual-hierarchy algorithm for calculating overall network congestion price had been demonstrated. In this scheme, the first hierarchy algorithm was set up in a central control server like Opendaylight and the guiding parameter B is obtained based on the intelligent data of global link state information. Based on the historical data, the congestion state of the network and the guiding parameter B is accurately predicted by the machine learning algorithm. The second hierarchy algorithm was installed in the Openflow link and the link price was calculated based on guiding parameter B given by the first algorithm. We evaluate this evolved NUMFabric algorithm in NS3, which demonstrated that the proposed NUMFabric algorithm could efficiently increase the link bandwidth utilization of cloud computing IoT datacenters.

KEYWORDS

Internet of Things; cloud computing; intelligent data aggregation; distributed optimization; trusted network calculation

1 Introduction

1.1 Background

The Internet of Things (IoT) was one of the main sources of big data [1]. By 2020, there will be more than 50 billion devices connected to each other [2], in addition to a large number of servers and routing devices, many edge devices and mobile devices are included [3], which will generate a large amount of data [4]. In addition to the security of IoT devices [5,6], it is essential to manage and transmit such a large amount of data due to big data often comes with a certain degree of insecurity [7]. Blockchain technology can figure out part of the problem of insecurity [8,9]. Data aggregation is an important method to solve the big data problem of the IoT [10]. The network of the IoT datacenter [11] was the kernel component of the cloud computing datacenter. It is responsible not only to support the interconnections for thousands of servers, but also provide high-rate data transmission for calculating services in the application layer [12]. The introduction of Software Defined Network (SDN) in the IoT can not only improve the flexibility and scalability of the IoT, but also provide a better solution for live data aggregation mechanism of IoT [13]. Since the mature detection schemes for network attacks against SDN was already presented, the security of SDN was guaranteed.

In the traditional Internet business, network traffic usually takes place between the client outside the IoT datacenter and the requested server inside the datacenter (It is called "north-south traffic"), where the traffic between the servers ("east-west traffic") was low [14]. In the emerging virtual-reality-calculation and online Internet business, large amounts of one-to-many or many-to-many communication between servers is required [15,16]. As a result, the internal traffic of cloud computing IoT datacenters grows rapidly and presents new characteristics different from the traditional Internet [17,18].

Different applications in the IoT datacenter have different bandwidth and latency requirements. For example, online query services such as responding to search engines are obviously more urgent than virtual machine migration or log backup. In order to satisfy the application's disparate demands with limited bandwidth, the server needs to decide how fast to send the packet. On the other hand, the link needs to determine the queue order of packets coming from different connections [19] and which route the packets should take if there are multiple paths [20]. The whole process depends on the congestion control, traffic scheduling and load balancing of the network [21]. Among them, network congestion deserves attention because it is easy for an attacker to seize the opportunity to launch a solar eclipse attack when the network is congested [22].

Recently, there have been a lot of research studied on transmission control protocol [20,23] in the networks of cloud computing IoT datacenters. A data-driven approach for Internet routing decision modeling in the future is also proposed [24]. These works have diverse targets for bandwidth sharing management. Besides minimizing the rate of deadline, latency or average flow completion time, other goals include multi-user bandwidth sharing. As a matter of fact, each improved transmission control protocol sustains one or two objectives in the bandwidth sharing strategy. However, for cloud computing IoT datacenters, it is more beneficial to adjust for diverse targets based on the user's demands [25].

Contrasted to the traditional networks which suffer from low bandwidth and long delay time, with a centralized design, the cloud computing IoT date center network is capable of providing massive traffic bandwidth, quick response and high throughput. It is worth noting that due to the difference in network architecture, to guarantee the data transmission performance, the transfer control algorithms used in traditional networks cannot be directly applied in the networks of cloud computing IoT datacenters.

The function model of multi-source and multi-link utility maximization was presented based on nonlinear programming theory in Jalaparti and Bliznets's Dynamic Pricing and Transmitting [25]. The key idea of such non-centralized and nonlinear programming theory is that the link prices are dynamically changing and the TCP sending windows are adjusting at a constant interval. Further, when the networking state is in balance, the networking utility will obtain maximization [26].

Until now, congestion pricing of the enhanced transmission control protocol for the current cloud computing IoT datacenter and the active queuing management approach in the link were involved in these theoretical nonlinear programming structures [27]. The important issues of network TCP congestion control are how to compute the link price according to the link status and how to regulate the data sending rate based on link congestion pricing feedback information. To address these issues, DCTCP [11,27], D2TCP [28], ICTCP [29], D3 [30], PDQ and DeTail DCTCP were put forward to satisfy the requirements of cloud computing IoT datacenter transmission control protocol. It is worth noting that these enhanced TCP's calculate the price of congestion link based on the ratio of network link statuses (for example the length of queuing, delay of packet queuing. length of packet queuing).

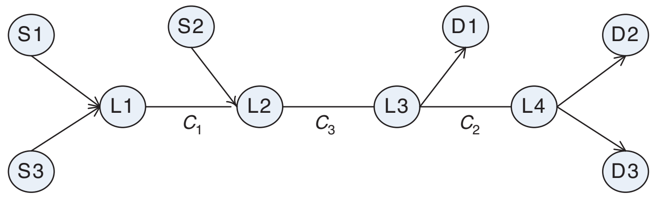

Figure 1: Multi-source and multi-link networking structure

In each cloud computing IoT datacenter network, the rates of transfer can be related by a utility function. The goal is to maximize network utility within the limit of (subject to) link capacity as expressed by

$$\max \sum_{i} U_i\left(x_i\right)$$

$$\textit{Subject to } Rx \leq c \tag{1}$$

where x is the rate vector of traffic TCP, R is the {0, 1} routing matrix (i.e., the element of R equals to 1 if and only if the TCP traffic i passes the respective link), and c is the link capacities vector. We assumed that the utility functions U(.) are smooth, increasing, and strictly concave. In [25,31], NUMFabric, an innovative transmission control protocol which provides nimble and quick bandwidth sharing control was proposed. Particularly, the proposed NUMFabric protocol is a decentralized protocol applied between the link end and TCP source. Further, the proposed NUMFabric protocol method can make the network converge on a certain specified equilibrium faster than any other previous protocol. The key idea of NUMFabric is based on the classic framework of non-linear programming maximization theory which tackles the TCP traffic flow resource-sharing problem by maximizing the utility function. This maximum utility function was created to benefit the diverse needs of link bandwidth sharing and could be updated by the network operator to accomplish different bandwidth and fairness intentions. Then the enhanced NUMFabric can enable the bandwidth sharing to ensure maximal total utility.

The main technical contribution of NUMFabric is a TCP congestion control which using the NUM method converges on balance faster than previous research work. The kernel idea of NUMFabric is decoupling the methods for network utility maximization and for maximum congestion link bandwidth sharing among common transport flows. The former NUM algorithms combine these goals and try to solve both of these problems at the same time via price changes at the links. This process is no longer robust because of the requirement to balance between the goal of speedy convergence towards optimal link bandwidth sharing and of avoiding a state of traffic jam or low-utilization. Although the NUMFabric achieved better performance than others, the method of calculating link price with the average residual can be improved upon using Openflow in the cloud computing IoT datacenter.

1.2 Main Idea

In this paper, the whole network structure has been obtained by the central control server in the SDN, and in order to calculate link price, we propose a kind of dual hierarchy algorithm based on non-linear programming. The idea of the upper layer method is applied in the kernel controller of the SDN (i.e., Opendaylight) network for the cloud computing IoT datacenter and the guiding parameter B is given by the improved NUMFabric method in the first layer based on all link statuses (net topological structure, state of link end and size of packet queuing are included). The second hierarchy method is applied in links and the web congestion signal price is calculated based on the guiding parameter B given by the first layer method.

1.3 Organization of the Rest of the Paper

The rest of this paper is organized as follows. We present the improved NUM framework for enhanced NUMFabric congestion signal calculation algorithm in Section 2. In Section 3, we present the detail for the dual-hierarchy network of enhanced NUM Fabric algorithm. Particularly, Subsection 3.1 introduces the first hierarchy algorithm in the central controller; the second hierarchy algorithm in Openflow link is proposed in Subsection 3.2. Scheme comparison is given in Section 4. Efficiency analysis and throughput verifiability analysis are given in Sections 5 and 6 respectively. At last, Section 7 concludes the paper.

1.4 Improved NUMFabric Algorithm General Framework

The improved NUM Fabric algorithm under the Software Defined Network structure is illustrated in Fig. 1. The kernel server is designed to implement the proposed dual-hierarchy networking congestion signal calculating method. The OpenDaylight in the kernel server comprises of the Abstract Service Layer (SAL) and basic networking services action. The basic networking services comprise of network structure control module, statistic control module, link control module, link forward rule management module, and host tracing module.

In the Software Defined Network structure, the information in the second hierarchy kernel link including new transport flow connection and network link status will be submitted to the top hierarchy method in the kernel central service controller. The network structure control module and statistic control module capture switch status information and propose theses link state operating information to the overall net status gathering module, so the first hierarchy method allocated in the kernel central controller in the cloud computing network can obtain real-time overall link state in web link. When the state of a small number of link load is low, the kernel service controller can change the routing flow table to balance the link load while the network transport flow will continue to send the packets. Therefore, it is not necessary to promote the link congestion price and restrain the transmitting rate of the sending TCP. The first hierarchy algorithm will not provide the radical guiding parameter B. But when all links are in peak load, it is out of the question to change the forward transmitting routing table. Therefore, in order to avoid all link states becoming the peak load state and to keep the cloud datacenter from becoming crowded, the kernel service controller will calculate the radical guiding parameter B and promote the link congestion signal price to restrict the sending window sizes of the source TCP.

2 The Hierarchical Link Price Calculation Algorithm

The key idea of the improved algorithm is that the first hierarchy algorithm in kernel controller in cloud computing network provides the guiding parameter B by applying the switch link congestion signal price calculation method based on all switch link state. When small link switch status are in lower load, the central controller in the cloud computing IoT datacenter can instruct the data packet to turn to the lower load. In this case, the first hierarchy algorithm will not provide the radical guiding parameter B. However, when all links are in peak load it is out of the question to change the forward transmitting routing table. Therefore, in order to avoid all link state becoming the peak load state and to keep the cloud IoT datacenter from getting crowded, the kernel service controller will calculate the radical guiding parameter B and promote the link congestion signal price to restrict the sending windows size of the source TCP.

2.1 *Upper Hierarchy Enhanced NUMFabric Method*

2.1.1 *All Openflow Link Running Status*

See Fig. 3, the first hierarchy algorithm in central control servers often sends the LLDP (Link Discovery Protocol PACKED) to all Openflow link and gathers the network structure information. Then Opendaylight in the central control servers can build the overall network structure. By using the protocol of Openflow among the central control servers and link switches, we modify the packet that contain server and link (both symmetric and asynchronous communication packets) and add new data to these packets' information structure. The modified data structure above contains the link status, guiding parameter B for the congestion signal calculation method and actual operating effect of the link in the cloud computing IoT datacenter.

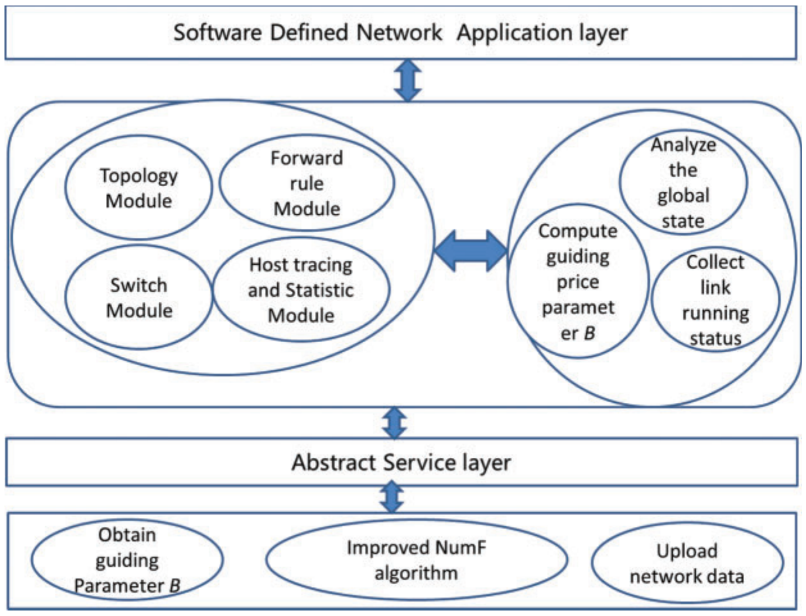
Software Defined Network Application layer
Topology Module
Forward rule Module
Switch Module
Host tracing and Statistic Module
Analyze the global state
Compute guiding price paramet er B
Collect link running status
Abstract Service layer
Obtain guiding Parameter B
Improved NumF algorithm
Upload network data

Figure 2: Improved NUM structure based on the software defined network in cloud computing IoT datacenters

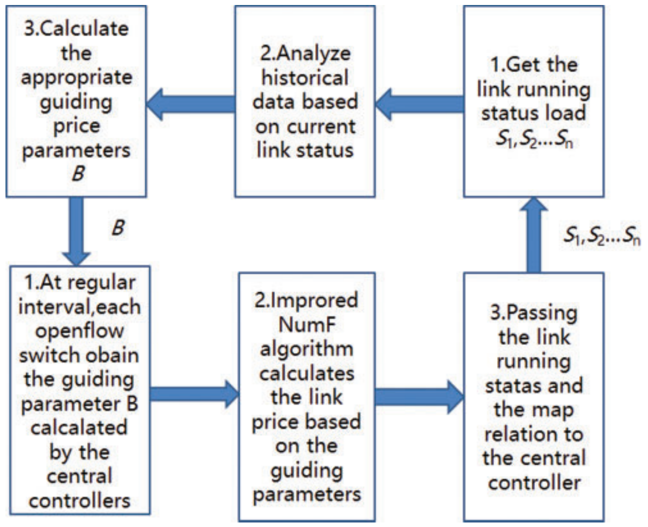
3.Calculate the appropriate guiding price parameters B
2.Analyze historical data based on current link status
1.Get the link running status load S1,S2...Sn
B
S1,S2...Sn
1.At regular interval,each openflow switch obain the guiding parameter B calcalated by the central controllers
2.Improred NumF algorithm calculates the link price based on the guiding parameters
3.Passing the link running statas and the map relation to the central controller

Figure 3: Dual hierarchy link price calculation algorithm of improved NUMF

2.1.2 Parameter Calculation

At the start time of the price calculation algorithm, based on the programming and openness of the Software Defined Network framework, the mapping relation among the network structure B of calculating link price parameter B and overall network link price are updated by the cloud computing IoT datacenter operator according to occupation history. When part Openflow link loads are light, the first hierarchy algorithm in central control servers give loose network structure B. But when all the links are under heavy load, the central control servers will provide the non-loose network structure B and promote the link congestion price to restrain the of the TCP source sending rate and avoid network congestion. The link load is $s_1, s_2, \ldots, s_n$, and the $B = \frac{1}{n}\sum_{i=1}^{n} S_i$.

2.2 Second Hierarchy Algorithm in Openflow Link

xWI is an innovative decentralized method for solving NUM problems; it operates with weighted max–min and realizes a transport layer like Swift. xWI is applied by iterative algorithm. For each iterating step, TCP flow communicates with the Openflow link to calculate the weights to set for their traffic flows in Swift. xWI iteratively obtains the weights to arrive at the balanced state of Network Utility Maximization. The important issues in xWI are to iteratively solve the KKT equations for the NUM problem.

2.2.1 System Model

As shown in Fig. 1, the link set is $L = \{1, \ldots, L\}$, and the capacity of link is c_l. $l \in L$.The source set is $S = \{1, \ldots, S\}$. Each source is described by four parameters $(L(S), U_s, m_s, M_s)$.

$L(s) \in L$ denotes that sources travel to the links set. U_s: $R_+ \to R$ is a utility function, m_s, M_s are the maximum and minimum transmitting rates of the source s. Set $I_s = [m_s, M_s]$ as the range of the sources rate x_s. Set $s(l) = \{s \in S \mid l \in L(S)\}$ as the set of sources travel to the link l. p_l is price of Openflow link.

The network utility optimization model is described as followed:

$$\max_{x_S \in I_S} \sum_{\mathrm{S}} U_S(x_s) \tag{2}$$

$$\sum_{S \in S(l)} x_s \leq c_l, \quad l = 1, \ldots, L$$

Define the Lagrangian function

$$L(x,p) = \sum_s U_S(x_S) - \sum_l p_l \left(\sum_{s \in S(l)} x_s - c_l \right)$$

$$= \sum_s \left(U_S(x_s) - x_s \sum_{l \in L(s)} p_l \right) + \sum_L p_l c_l \tag{3}$$

The dual problem of (2) is described as follows:

$$D(p) = \max_{x_s \in I_s} L(x,p) = \sum_S B_S\left(p^S\right) + \sum_l p_l c_l$$

where

$$B_s p^s = \max_{x_s \in I_S} U_s x_s - x_s p^s \tag{4}$$

$$p^s = \sum_{l \in L(s)} p_l$$

We can obtain the dual problem as follows:

$$D: \min_{p \leq 0} D(p) \tag{5}$$

Then $D(p) = \sum_s \left(U_s(x_s(p)) - x_s(p)\, p^s \right) + \sum_l p_l c_l$

where U_s is strictly concave, $D(p)$ is continuously differentiable with derivatives given by

$$\frac{\partial D}{\partial p_l}(p) = c_l - x_l(p) \tag{6}$$

where $x_l(p) = \sum_{s \in S(l)} x_s(p(t))$ is the aggregate source rate at link l.

Substituting (4) to (6) we obtain the following link price adjustment rule for link $l \in L$:

$$p_l(t+1) = p_l(t) + \gamma\left(x_l(p(t)) - c_l\right) \tag{7}$$

At the time of $t+1$, the prices of Openflow link are modified based on the values in iteration t. The modified rule contains two terms, corresponding to the two optimality conditions. The first part is obtained by

$$p_l^{res} = p_l(t) + \min_{i \in S(l)} \left(\frac{U\prime(x_i(t)) - \sum_{K \in L(i)} p_k(t)}{|L(i)|} \right) \tag{8}$$

where $L(i)$ is the number of links in flow i's path.

The second part based on underutilization form is given by

$$p_l^{new} = p_l^{res} + \eta \left(1 - \frac{\sum_{i \in S(l)} x_i(t)}{c_l} \right) p_l(t) \tag{9}$$

(1) For each constant time interval, the central control servers will acquire map of the relationship between the link running status and vector B.
(2) As shown in Fig. 2, vector B is from the first hierarchy algorithm, which is based on calculating the price of Openflow link of each link.

$$p_l^{res} = p_l(t) + \frac{s_l(t)}{B(t)} \min_{i \in S(l)} \left(\frac{U_i'(x_i(t)) - \sum_{K \in L(i)} p_k(t)}{|L(i)|} \right) \tag{10}$$

The second algorithm in the openflow switch:

At time $t = 1, 2, \ldots,$ in link l

Step 1: Receives rates $x_s(t)$ from all sources $s \in S(l)$ that travel through link l.

Step 2: Calculates a new link price

$$p_l^{new}(t+1) = p_l^{res} + \eta \left(1 - \frac{\sum_{i \in S(l)} x_i(t)}{c_l} \right) p_l(t)$$

This is the refinement calculation method for link price.

Combining these two terms, we obtain as follows:

$$p_l(t+1) = \beta p_l^{res} + (1-\beta) p_l^{new}(t)$$

Here, $\beta \in (0, 1)$ is the filtering parameter. (Set to 0.5 in our experiment).

Step 3: Communicate new price to all sources $s \in S(l)$ that travel through link l.

Source algorithm:

Step 1: Receives rates $x_s(t)$ from all sources $s \in S(l)$ that travel through link l.

Step 2: Chooses a new transmission rate $x_s(t+1) = x_s(t)(p^s(t))$

Step 3: Communicate new rate $x_s(t+1)$ to links $l \in L$ in its path.

The calculation results are shown in Tab. 1.

The improved NUMF method makes full use of the trait that the core server control in SDN can obtain the global link status and calculate the guiding parameter B. Compared with related methods, the link calculation price rule is divided into two terms and therefore more refined.

The update rule for calculating price consists of two terms, namely formula (8) and formula (10), corresponding to the two optimality conditions.

Table 1: The link load and the rate parameter B

No.	s1	s2	s3	s4	Load	B	s1/B	s2/B	s3/B	s4/B
1	0.2	0.3	0.2	0.3	Lighter	0.25	0.8	1.2	0.8	1.2
2	0.4	0.5	0.7	0.6	Light	0.55	0.72	0.9	1.27	1.11
3	0.7	0.6	0.8	0.5	Low heavy	0.65	1.07	0.92	1.23	0.78
4	0.8	0.9	0.7	0.6	Heavy	0.75	1.06	1.2	0.94	0.8

3 Related Method Comparison

We design a semi-dynamic scenario to quantify the throughput per-flow. In the semi-dynamic case, we can trigger network incident in a controlled method and meter the network throughput.

We make the simulation at a cloud computing IoT datacenter network which is built using a leaf-spine structure with NS3. There are 32 servers connected to 4 leaf openflow links and the link bandwidths is 1000 Mbps. Each leaf openflow link is connected to 4 spine links and the link bandwidth is 10 Gbps, so it can guarantee full bisection bandwidth. The openflow links are intended to be standard output-queued links, with a buffer of size 1 MB per port. We have compared the convergence time with the following method.

3.1 FAST TCP

Fast TCP optimizes TCP traffic over wide area networks and wireless data networks, especially in TCP environments with high latency and packet loss. Fast TCP does not change the standard format of TCP Packet Header, but the traffic control algorithm is optimized, which greatly improves the efficiency of TCP traffic and the utilization of WAN bandwidth.

3.2 DCTCP

We will extensively deploy TCP's congestion control algorithm like DCTCP. DCTCP adopts Explicit Congestion Notification (ECN) in switches to detect and respond to network congestion signal by sequencing ECN marks via the switch [32]. DCTCP provides the same or better throughput than TCP, while reducing the router buffer space by 90% and ensuring security. In the link, DCTCP adopts a very simple active queue management mechanism. When the queue takes up more than a certain threshold value k, the arriving packet is marked with the CE (Capacity Experience) flag. On the DCTCP receiving end, DCTCP returns the ECN and accurately conveys which packets have experienced congestion.

3.3 Improved NUM

This improved NUMFabric method propose a dual hierarchy method. The first hierarchy algorithm in the kernel controller produce the guiding parameter B for calculating switch link congestion signal price by taking full advantage of all switch link state information. The update rule for calculating price in the second hierarchy method consists of two terms, corresponding to the two optimality conditions.

4 Experimental on NS3

In this section, according to evaluation of improved NUMFabric, we give a NS3 simulation. The goal is to simulate the throughput of NUMFabric to maximize the allocation in dynamic settings and to achieve bandwidth allocation goals precisely and robustly. We choose one-to-one communication, where our server communicates randomly with another server once a time. The results are demonstrated in Figs. 5 and 6.

Convergence: Fig. 4 displays the CDF convergence time of FAST TCP, DCTCP, and improved NUMFabric. The average convergence time of Improved NUMFabric is 591 (μs). The average convergence time of FAST TCP is 726 (μs). And the average convergence time of DCTCP is 813 (μs).

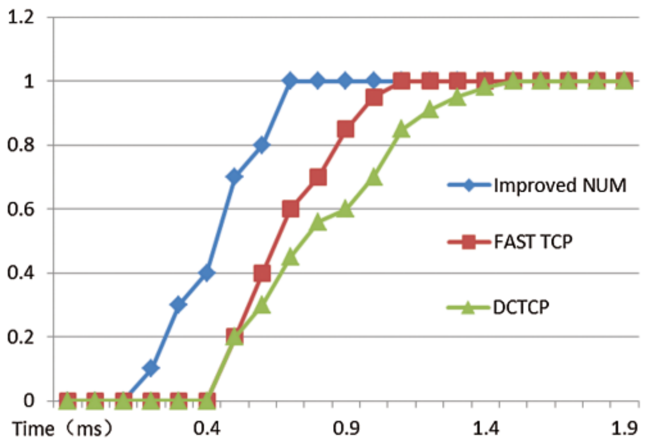

Figure 4: CDF of convergence time for improvedNUM, FAST and DCTCP

The main reason for the improvement in NUMFarbic convergence speed is that the link price is calculated by the global guiding parameter B which takes into consideration the global network running status. Further, the update rule for price calculation was also more refined designed; it made up with two terms, each according with two optimum conditions. From above CDF of convergence time, we can see that the improved NUMFabric achieves better convergence performance.

4.1 Rate Stability Comparison

Fig. 5 displays the rate achieved by a typical FAST TCP flow during several network events. The expected rate of the FAST flow is shown by the blue line connecting the datapoints. It is clear from Fig. 5 that the FAST TCP flow does not ever converge to within 10%.

Fig. 6 displays the rate reached a representative DCTCP flow when few network-events happen. The anticipant rate of FAST flow is given from the blue one. that the Fig. 6 depicted syllabify that the DCTCP flow does not ever converge to within 10%.

Compared to Figs. 5–7 displays a better rate reached by the according flow the Improved NUMFabric.

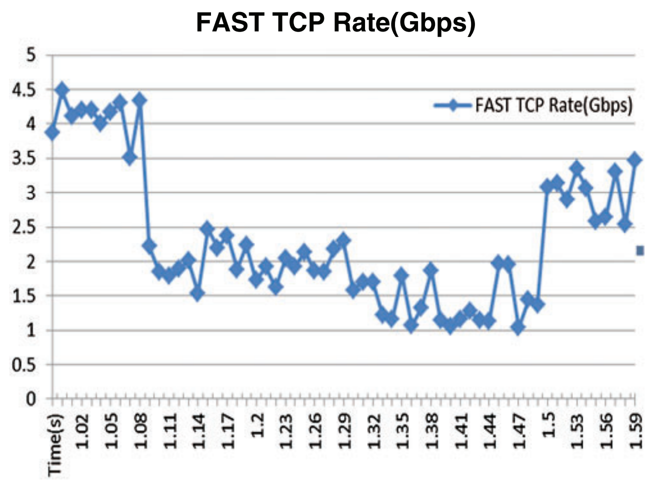

Figure 5: Rate of FAST TCP

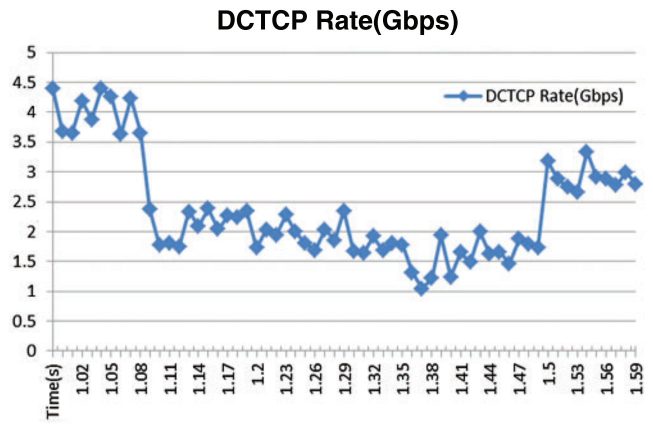

Figure 6: Rate of DCTCP

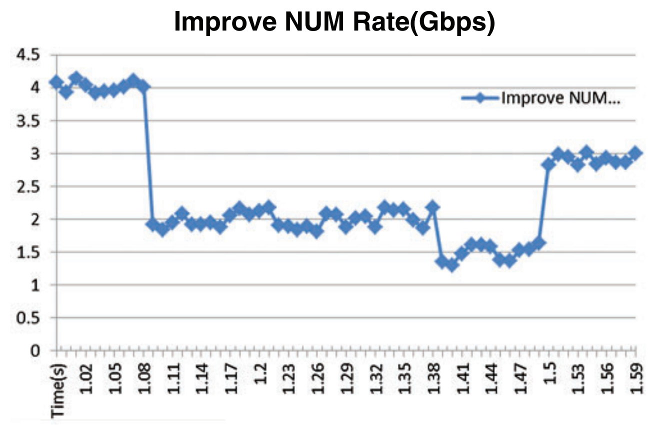

Figure 7: Rate of Improved NUMFabric

5 Conclusion

The IoT data aggregation scheme based on SDN designed in this paper can improve NUM-Fabric algorithm for calculating overall congestion signal. We get the most advantage out of the trait that the whole network structure has been obtained by the central control server in the Software Defined Network and proposed a kind of dual hierarchy method for calculating overall network congestion signal. The first hierarchy method is set up in a central control server like Opendaylight and obtains the guiding parameter B based on the overall link state information. The second hierarchy method is assigned in Openflow link and the link price is calculated based on guiding parameter B given by the first method. The update rule for calculating price in the second hierarchy method consists of two terms, which correspond to the two optimality conditions. The simulation results demonstrate that this improved NUMFabric method can indeed provide better rate stability, reduce the time delay of data aggregation, improve the accuracy of data aggregation and the performance of the network.

Funding Statement: The authors thank all the reviewers and editors for their valuable comments and works. This work is supported by National Key R&D Program of China—Industrial Internet Application Demonstration-Sub-topic Intelligent Network Operation and Security Protection (2018YFB1802400).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Qiu, J., Tian, Z., Du, C., Zuo, Q., Su, S. et al. (2020). A survey on access control in the age of Internet of Things. *IEEE Internet of Things Journal, 7(6),* 4682–4696. DOI 10.1109/JIOT.2020.2969326.
2. Cheng, Z., Wang, Z., Luo, Z. (2019). Dynamic fracture analysis for shale material by peridynamic modelling. *Computer Modeling in Engineering and Sciences, 118(3),* 509–527. DOI 10.31614/cmes.2019.04339.
3. Tian, Z., Luo, C., Qiu, J., Du, X., Guizani, M. (2019). A distributed deep learning system for web attack detection on edge devices. *IEEE Transactions on Industrial Informatics, 99,* 1. DOI 10.1109/TII.2019.2938778.
4. Cheng, Z., Jin, D., Yuan, C., Li, L. (2019). Dynamic fracture analysis of functionally gradient materials with two cracks by peridynamic modeling. *Computer Modeling in Engineering & Sciences, 121(2),* 445–464. DOI 10.32604/cmes.2019.06374.
5. Du, X., Zhang, M., Nygard, K. E., Guizani, S., Chen, H. H. (2007). Self-healing sensor networks with distributed decision making. *International Journal of Sensor Networks, 5(6),* 289–298. DOI 10.1504/IJSNET.2007.014354.
6. Du, X., Guizani, M., Xiao, Y., Chen, H. H. (2008). Defending DoS attacks on broadcast authentication in wireless sensor networks. *2008 IEEE International Conference on Communications*, pp. 1653–1657, Beijing, China. DOI 10.1109/ICC.2008.319.
7. Li, M., Sun, Y., Lu, H., Maharjan, S., Tian, Z. (2020). Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems. *IEEE Internet of Things Journal, 7(7),* 6266–6278. DOI 10.1109/JIOT.2019.2962914.
8. Wu, L., Du, X., Wang, W., Lin, B. (2018). An out-of-band authentication scheme for Internet of Things using blockchain technology. *2018 International Conference on Computing, Networking and Communications*, pp. 769–773, Maui, HI. DOI 10.1109/ICCNC.2018.8390280.
9. Tian, Z., Li, M., Qiu, M., Sun, Y., Su, S. (2019). Block-def: A secure digital evidence framework using blockchain. *Information Sciences, 491(2019),* 151–165. DOI 10.1016/j.ins.2019.04.011.

10. Honarvar, A. R., Sami, A. (2016). Extracting usage patterns from power usage data of homes' appliances in smart home using big data platform. *International Journal of Information Technology and Web Engineering, 11(2),* 39–50. DOI 10.4018/IJITWE.2016040103.
11. Alizadeh, M., Greenberg, A., Maltz, D. A., Padhye, J., Pater, P. et al. (2010). Cloud computing datacenter TCP (DCTCP). *Proceeding of the SICOMM,* pp. 156–163, New Delhi, India.
12. Chang, S. H., Mao, S. H. (2016). A novel software-defined wireless network architecture to improve ship area network performance. *Journal of Supercomputing, 73(7),* 3149–3160. DOI 10.1007/s11227-016-1930-5.
13. Tian, Z., Gao, X., Su, S., Qiu, J., Du, X. et al. (2019). Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory. *IEEE Transactions on Vehicular Technology, 68(6),* 5971–5980. DOI 10.1109/TVT.2019.2910217.
14. Xu, H., Li, B. (2014). Repflow: Minimizing flow completion times with replicated flows in data centers. *IEEE INFOCOM 2014—IEEE Conference on Computer Communications*, pp. 1581–1589. Toronto, Canada. DOI 10.1109/INFOCOM.2014.6848094.
15. Xu, H., Li, B. (2012). Universally compostable zero-knowledge sets. *Inderscience Journal of Grid and Utility Calculating, 3(1),* 25–31. DOI 10.1504/IJGUC.2012.045695.
16. Yin, L., Luo, X., Zhu, C., Wang, L., Xu, Z. et al. (2020). Connspoiler: Disrupting C&C communication of iot-based botnet through fast detection of anomalous domain queries. *IEEE Transactions on Industrial Informatics, 16(2),* 1373–1384. DOI 10.1109/TII.2019.2940742.
17. Hong, C., Caesar, M., Godfrey, P. B. (2012). Finishing flows quickly with preemptive scheduling. *Proceeding of the SICOMM,* pp. 231–239, Helsinki, Finland.
18. Huang, X., Du, X. (2014). Achieving big data privacy via hybrid cloud. *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 512–517. Toronto, Canada. DOI 10.1109/INFCOMW.2014.6849284.
19. Qiu, J., Du, L., Zhang, D., Su, S., Tian, Z. (2019). Nei-TTE: Intelligent traffic time estimation based on fine-grained time derivation of road segments for smart city. *IEEE Transactions on Industrial Informatics, 16(4),* 2659–2666. DOI 10.1109/TII.2019.2943906.
20. Zhang, H., Chen, L., Yi, B., Chen, K., Chowdhury, M. et al. (2016). Toward automatically identifying and scheduling coflows in the dark. *Proceedings of the SICOMM,* pp. 233–246, Rio De Janeiro, Brazil.
21. Tian, Z., Gao, X., Su, S., Qiu, J. (2019). Vcash: A novel reputation framework for identifying denial of traffic service in internet of connected vehicles. *IEEE Internet of Things Journal, 7(5),* 3901–3909.
22. Tan, Q., Gao, Y., Shi, J., Wang, X., Fang, B. et al. (2019). Toward a comprehensive insight into the eclipse attacks of tor hidden services. *IEEE Internet of Things Journal, 6(2),* 1584–1593. DOI 10.1109/JIOT.2018.2846624.
23. Du, X., Shayman, M., Rozenblit, M. (2001). Implementation and performance analysis of SNMP on a TLS/TCP base. *IEEE/IFIP International Symp. on Integrated Network Management*, pp. 453–466, IEEE.
24. Tian, Z., Su, S., Shi, W., Du, X., Guizani, M. et al. (2019). A data-driven method for future internet route decision modeling. *Future Generation Computer Systems, 95,* 212–220. DOI 10.1016/j.future.2018.12.054.
25. Huang, H., Xu, L. (2016). Design and analysis of the secure scheme for quantum positioning based on entangled photon pair. *International Journal of Technology and Human Interaction, 12(2),* 22–35. DOI 10.4018/IJTHI.
26. Huang, H., Guo, S., Li, P., Liang, W., Zomaya, A. Y. (2016). Cost minimization for rule caching in software defined networking. *IEEE Transactions on Parallel & Distributed Systems, 27(4),* 1007–1016. DOI 10.1109/TPDS.2015.2431684.
27. Zhang, J., Zhang, F. (2013). Linear threshold verifiable secret sharing in bilinear groups. *Inderscience Publishers, 4(2/3),* 212–218. DOI 10.1504/IJGUC.2013.056258.
28. Wang, Y., Ma, J., Lu, X., Lu, D., Zhang, L. (2016). Efficiency optimisation signature scheme for time-critical multicast data origin authentication. *International Journal of Grid & Utility Computing, 7(1),* 1–11. DOI 10.1504/IJGUC.2016.073771.
29. Wu, H., Feng, Z., Guo, C., Zhang, Y. (2012). ICTCP: Incast congestion control for TCP in data-center networks. *IEEE/ACM Transactions on Networking, 21(2),* 345–358. DOI 10.1109/TNET.2012.2197411.

30. Wilson, C., Ballani, H., Karagiannis, T. (2011). Better never than late: Meeting deadlines in cloud computing datacenter networks. *Proceedings of the SICOMM,* pp. 332–339, Toronto, Canada.

31. Nagaraj, K., Bharadia, D., Mao, H., Chinchali, S., Alizadeh, M. (2016). NUMFabric: Fast and flexible bandwidth allocation in cloud computing datacenters. *Proceedings of the SICOMM,* pp. 188–201, Rio De Janeiro, Brazil.

32. Tian, Z., Shi, W., Wang, Y., Zhu, C., Du, X. et al. (2019). Real-time lateral movement detection based on evidence reasoning network for edge computing environment. *IEEE Transactions On Industrial Informatics/A Publication of the IEEE Industrial Electronics Society, 15(7),* 4285–4294. DOI 10.1109/TII.2019.2907754.